Automated Cell Based Generalization of Virtual 3D City Models with Dynamic Landmark Highlighting

Tassilo Glander and Jürgen Döllner
Hasso-Plattner-Institute (University of Potsdam)
{tassilo.glander, doellner}@hpi.uni-potsdam.de

Current visualization of virtual 3D city models does not deal with abstraction to facilitate easy comprehension of the displayed data. Therefore, we present a technique to automatically generalize virtual 3D city models and create a visualization that dynamically exaggerates global landmark objects. Extending earlier work, our technique uses the infrastructure network to cluster buildings in cell blocks, while preserving local landmarks in their representation. A landmark hierarchy is created based on these initially found landmarks. As the generalization process creates representations of higher levels of abstraction, only landmark buildings from the higher levels of the landmark hierarchy are preserved, effectively reducing their number for clarity. Finally, our real-time visualization exaggerates the most important global landmarks dynamically by scaling them.

1 Introduction

As technologies for remote sensing, modeling and storing geospatial data get increasingly sophisticated, also virtual 3D city models get more and more detailed and cover whole city areas, containing several hundreds of thousands of building models. The growing geometrical and visual complexity leads to fundamental problems for their visualization: Perceiving large scale high detailed virtual 3D city models requires the dedicated attention of the user, since there are too many heterogeneous objects to pre-attentively recognize them and to quickly identify urban structures. In addition, due to the perspective, objects far away from the camera are depicted very small. Thus, they cannot be differentiated and their individual display is of limited use.

For the creation of maps, limited print resolution and space always required generalization of the spatial structures and relations to be depicted. The identified and established visualization principles of cartographic generalization need to be transferred to 3D geovisualization, as well as existing solutions may be applied in 3D, too. Additionally, new challenges have to be dealt with, some of them might be solved by generalization:

- Massive data sets have to be visualized, which potentially involves generalization processes. These have to maintain characteristic patterns.
- In 3D perspective, forefront objects occlude background objects. While this reflects real-life experience, it hides structures that might be of value, like a route destination.

  In addition, it leads to a continuous range of scales being depicted in a single image, however, it is not clear how to present this.
- User interaction allows alteration of the depicted situation, e.g., by moving and zooming the virtual camera or by changing other parameters of the visualization. The transition between different states has to be coherent to avoid user disorientation.
- When using different degrees of abstraction in one image, this must be communicated to the user. Otherwise, the user's mapping between the displayed model and reality will be based on wrong assumptions.

In this paper, we address some of these challenges. We integrate two formerly separate techniques, namely the cell based generalization technique as in [13, 14] and the landmark visualization from [15]. The cell based generalization yielded a static visualization and did not make use of the interactivity of the 3D visualization. Since 2D maps often present landmarks in different exaggerated size in smaller scales, this concept is familiar with the user and should be transferred to 3D as well. We are implementing that in this paper as a proof of concept. In addition we adapt the landmark selection process used during the generalization process to create a landmark hierarchy as suggested by Winter et al. in [29]. We map the hierarchy levels to the LOD levels calculated and use the landmarks accordingly in the generalization.

2 Related Work

Map generalization techniques in 2D are implementing generalization operators [21, 16] to automatically derive maps for smaller scales. These techniques use approaches from artificial intelligence, e.g., agent systems [3, 20] or artificial neural networks [1], as well as optimization methods, e.g., least squares adjustment [17, 24], simulated annealing [27], or force models [4, 5].

Specifically seeking simplification of 3D building models, a number of approaches have been researched in the last couple of years. It has been recognized, that building structures need a different simplification approach than generic 3D model simplification techniques, such as surveyed in [18] and [8]. Specific properties such as parallel and orthogonal walls need to be respected and even enforced during simplification. Thiemann and Sester apply a feature removal technique on 3D building models to iteratively extract small shapes from the model, and store them in a constructive solid geometry (CSG) representation. Using the CSG tree, the appropriate shapes for the desired level of detail can be selected for display [25]. Kada suggests a technique that remodels the building by a few characteristic planes. Coplanar, parallel and orthogonal features are preserved and enforced, also roof geometry is simplified [19]. Forberg presents an approach that moves close parallel planes until they merge to remove small protrusions and close

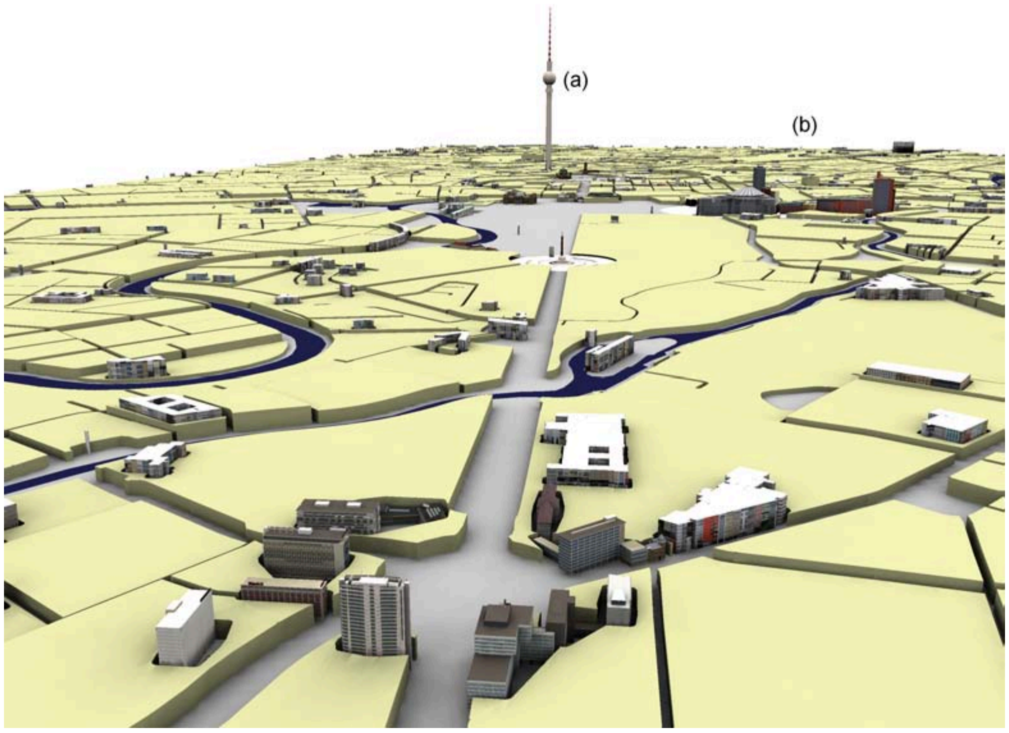

Figure 1: The result of the generalization yields an abstract visualization: Simplified block representations replace buildings with less importance. Only buildings detected as local landmarks are still shown with original facade textures. In addition, global landmarks as the TV tower (a) or the *Potsdamer Platz*(b) are exaggerated in their scale.

gaps. It requires an orthogonalization before to yield good results [11]. Rau et al. use roof and wall collapse operations dedicated on their 3D building representation consisting of polyhedra [23].

For large scale city models, single building simplification alone does not suffice for visualization. Typically, the individual buildings are geometrically simple except a small percentage of landmark buildings. Instead, the huge number of single objects contributes most to both computational complexity and cognitive load. Traditional simplification algorithms fail to reduce the number of points in this case, as they do not simplify the genus. A building generalization technique working for linear building groups is shown by Anders [2]. Based on well working 2D generalization algorithms, it is also capable of aggregating building models. Coors applied a modified version of [12] to aggregate common buildings while preserving and enhancing important ones [10].

An idea similar to our approach is presented by Chang et al. [6, 7]. They cluster buildings according to their distance, compute a hull for each cluster, and simplify the hull. Then, the weighted average is applied as the cluster's height, buildings differing dramatically from the average height are regarded as landmarks and kept in the cluster. The authors, however, focus on photo-realistic visualization and use the created blocks to apply realistic textures.

It can be summed up that building simplification is addressed by a number of publications, though approaches for aggregation are rare. However, to generalize a whole city model for visualization of smaller scales, aggregation becomes more important, since the overall complexity is mainly owed to the mere number of objects.

3 Concept

In this section, we will give a brief overview of our technique. Our goal is a visualization that presents urban space simplified, yet provides sufficient information. By sufficient information we mean, that the user is able to orient and navigate easily within the virtual city model.

When perceiving a photo realistic representation of a city model, the user has to deal with a large amount of information. Depending on the task to be done, displaying the data in its entirety may be unnecessary or even distracting. Explorative interaction with the model like zooming-in is a way to resolve information density, e.g., when the user looks closely at street-level details. However, in an overview perspective, many users have problems identifying even known cities. Moreover, orientation and navigation is nontrivial for untrained users, since necessary information such as typical road network structures, central landmarks, and overall shape are hidden in the visual noise. The noise is resulting from textural information as well as the mere number of heterogeneous single features, such as buildings.

Therefore, we suggest a visualization mimicking the style of 2D maps: The city is presented as consisting of simple block cells, putting the focus on the structuring infrastructure network. Only landmark buildings remain unchanged in the city model, as they represent essential orientation marks. The generalized model is enhanced with depth cues provided by shadow textures applied (Fig. 1).

Our technique creates a number of representations with growing abstraction, named *level of abstraction* (LOA) for distinction from single building's level of detail (LOD as in CityGML [9]). In each LOA, the block cells get larger, aggregating more buildings. Also, the number of landmarks is decreasing during the abstraction, since local landmarks loose their importance in smaller scales. In terms of the CityGML LODs, our technique uses buildings from LOD1-4 as input and creates abstract representations that belong somewhere below LOD1, but still contain buildings (landmarks) with a higher LOD.

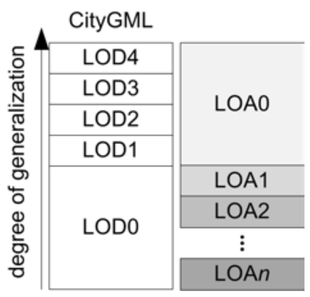

Figure 2: CityGML LOD vs. LOA

We base the landmark selection in higher LOAs on Winter et al. [29], who discuss landmarks in the different contexts of way finding and spatial knowledge. They conclude that there are local landmarks and different levels of global landmarks, differentiated by the size of their reference region. Additionally, they describe an algorithm to derive a landmark hierarchy mapping landmarks to different levels of salience. This fits very well to our demands and we use their algorithm to select landmarks for our different LOAs, improving our former approach.

In addition to displaying these abstracted, but static models, we are using the capability of in-

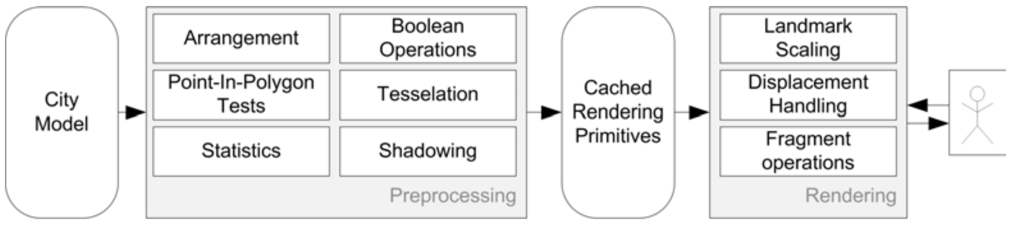

Figure 3: The visualization pipeline shows, how the input city model data is first generalized in a preprocessing stage and later rendered to yield an interactive, user explorable visualization.

teractive visualization to further highlight a limited number of the most global landmark buildings. The landmarks set on the top of the calculated hierarchy are selected as *dynamic landmarks*, having importance for large areas of the city. These central landmarks are displayed enlarged, with the scaling factor being adapted dynamically depending on the camera distance. Therefore we call them *dynamic* landmarks.

In a real city, directional markers give a position a contextual reference frame; they not only give directions for people going to these places, but also help positioning between the landmarks. Our visualization ensures the visibility of landmarks, thus supporting the development of survey knowledge. In addition, we use the landmarks themselves instead of abstract symbols to show their positions, which is preferable [26].

To implement such a visualization, our technique has two parts, the first being a preprocessing stage, and the second being the interactive rendering stage at runtime (Fig. 3). In the first part, the city model geometry is processed: The input data consisting of shape files (buildings and streets) and CAD models are used to create a generalized representation (LOA1). Additionally, a landmark hierarchy is created from the initial set of detected landmarks. In the course, a number of subsequent representations with growing level of abstraction (LOA2-n) is calculated, each integrating the landmarks from the appropriate hierarchy level. Finally, the resulting geometry is converted to rendering primitives and the shadow textures are pre-calculated. In the second stage the rendering displays one LOA representation. The landmarks from the highest layer of the landmark hierarchy are selected for the dynamic highlighting. For those, a scaling factor is calculated for every rendered frame, based on the individual camera distance.

4 Preprocessing

The first part of the generalization technique reads the input data and creates a number of generalized representations (LOAs). It has to be done once.

4.1 Data Handling

As input data, our technique needs buildings and an infrastructure network. Virtual 3D city models typically contain a large number of geometrically relatively simple, automatically created building models from cadastral databases or aerial laser scanning, and a small number of high detail CAD models (LOD3-4) created manually. Thus, our technique processes simple prismatic block buildings (LOD2), e.g., as in shape files with a height attribute, and detailed CAD models, e.g., as geo-positioned 3DS, collada or X3D files. The infrastructure network needs to be present as weighted linear features. This is the case with TeleAtlas® or OpenStreetMap data. Additionally, land use data can be added for lakes, rivers, forests, and green spaces.

CAD models are projected to their footprints during the preprocessing. This provides a homogeneous access to all of the input data, while the footprints are still referenced with the original models for later use in the rendering part. We set up a simple data model for features taking part in the generalization process (Fig. 4). In addition to the footprint geometry, all features have at least an `ID` and a `generalizesToID` to track them between different levels of generalization.

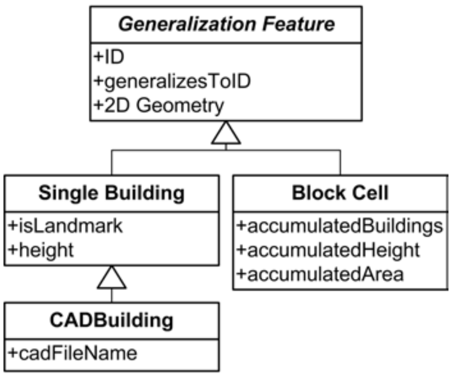

Figure 4: Features are distinguished as being single buildings or block cells during the generalization process. The generalizes-to relation is tracked for all features and intermediate results are stored and / or modified.

4.2 Creating Generalized Block Cells

As in [13, 14], the road network is used to segment the plane into polygonal cells using the arrangement package from the Computational Geometry Algorithms Library (CGAL) [28]. The building footprints are then mapped to the computed cells using point-in-polygon tests and both the accumulated height and the height variance is calculated for each cell. The values for accumulated height, accumulated footprint's area, and the number of contained buildings are stored as attributes of the cell polygons. In addition, the contained buildings store the `ID` of the cell that generalizes them. For one cell, local landmark buildings are identified by comparing the individual height with the mean height of the cell. We mark buildings b_i of a cell i with a height considerably above the mean height $\overline{h}$ of that cell as landmarks. We describe "considerably above" in k units of the standard deviation σ, typically with $k = 2$.

$$\mathrm{isLandmark}(b_i) = \mathrm{height}(b_i) > k \cdot \sigma + \overline{h} \qquad (1)$$

This forms the initial set of landmarks.

In the following, the landmarks' footprints as well as water areas and the buffered roads are subtracted from the cell polygons to further shape the blocks. The roads are buffered according to their weight, i.e., important roads are wider than less important ones.

This process is done for the first level of abstraction (LOA1). In higher levels, it is repeated with fewer streets, iteratively dropping the streets with the lowest weight, until no streets are left. Thus, the cells get larger and more features are aggregated in one block. However, the landmarks in the higher levels are determined differently, as pointed out in the next section.

4.3 Deriving a Hierarchy Based on Height

In our previous implementation [13, 14], we used the above selection function for all levels of abstraction. To yield fewer landmarks, we set the criteria stricter by setting higher values for k. However, that involved manual tweaking of the parameter and depended on the dataset used.

In this implementation, we used the technique of Winter et al. [29] to create a landmark hierarchy. The centroids of the initially found landmark buildings are used as input for a Delaunay triangulation. Again, we use the according CGAL package [22] to perform the triangulation. Loosely following the algorithm of Winter et al., for each vertex of the triangulation one successing vertex is voted for. The voting is done by choosing from the vertex and the vertex' topological neighborhood the one with the highest salience value. All vertices that have at least one vote are promoted to the next hierarchy level and form again a Delaunay triangulation.

We implemented the voting as a loop over all vertices and through all neighbors, exposing the comparing method to the caller (see Fig. 5 for pseudo code). Thus, the algorithm can be easily extended to evaluate other attributes of the buildings, e.g., visual information acquired before. Lacking other measures for salience, we implement the `lessThan` relation by comparing the height.

As a result, the number of vertices and the corresponding landmark buildings is steadily reduced in subsequent layers of the hierarchy. In average, we found a reduction factor of around 3. The algorithm stops, when just one landmark is left. The great advantage of this method is, that the iterative elimination of landmarks in higher hierarchy levels maintains an even spatial distribution. In addition, it requires no manual interaction.

We align the calculated landmark hierarchy to the levels of abstraction by using the initial landmark set for the first generalized level (LOA1) and the higher levels of the hierarchy for the following generalization levels. Thus, all other generalization levels are created using the landmark hierarchy.

4.4 Preparing Rendering Primitives

The resulting 2.5D geometry is then prepared for rendering: The footprints are extruded to simple blocks using the stored height for buildings and the calculated mean height for generalized blocks. In the case of landmark buildings, the facade texture is applied as well. Through tessellation, rendering primitives for the graphics hardware are created. For the footprints of CAD models, the referenced 3D models are loaded and added to the scene. While preparing the rendering primitives, the `ID` and `isLandmark` attributes are set as per-vertex attributes, exposing the graphics hardware to evaluate them, later. The resulting scene is then pre-rendered to yield

```
Triangulation[] createTriangulationHierarchy(Triangulation t, &lessThan) {

        Triangulation[] hierarchy;
        hierarchy.append(t);
        Triangulation currentTriangulation = t;

        while(currentTriangulation.number_of_vertices() > 1){
                Triangulation newTriangulation

                foreach(Vertex v in currentTriangulation){
                        Vertex vote = v //initialize vote

                        // go through all neighboring vertices
                        Vertex firstNeighbor = v.nextNeighbor()
                        Vertex neighbor = firstNeighbor
                        do{
                                // call external callback to expose
                                // the less than relation
                                if( lessThan(vote,neighbor) ){
                                        vote = neighbor
                                }
                                neighbor = v.nextNeighbor()
                        }while(neighbor != firstNeighbor)

                        if ( !newTriangulation.contains(vote) ){
                                newTriangulation.add(vote)
                        }
                }
                hierarchy.append(newTriangulation);
                currentTriangulation = newTriangulation

        }
        return hierarchy;
}
```

Figure 5: Pseudo code for creating the triangulation hierarchy. The `lessThan` callback exposes the compare operation to the caller. We implemented it comparing the height of the associated landmark buildings.

shadow textures by applying ambient occlusion, a global illumination approach. Finally, the scene can be stored on disk for fast reloading later.

5 Rendering

The visualization displays the static geometry as in [13], but is enhanced in this implementation with the dynamic highlighting of the most important landmark objects. We constrain the number of dynamic landmarks to ten to avoid getting the scene filled and the user's attention overloaded.

The prepared scene is at first traversed to identify the dynamic landmarks. The positions and extents of the dynamic landmarks are collected and stored as global variables available for all geometry.

5.1 Dynamic Highlighting

At runtime, the dynamic landmarks have to be scaled according to their distance to the viewer. A scaling function $\mathrm{scale}_i(d)$ has to be defined [15]. To intuitively describe the properties of the scaling, we choose a distance interval $I = [d_{\mathrm{start}} : d_{\mathrm{end}}]$. The depicted size of the scaled landmark should remain (nearly) constant when zooming out from the starting distance, until the end distance is reached and the landmark returns to its original size (Fig. 6). For a smooth transition we use a quadratic function and calculate its coefficients depending on the interval I [15].

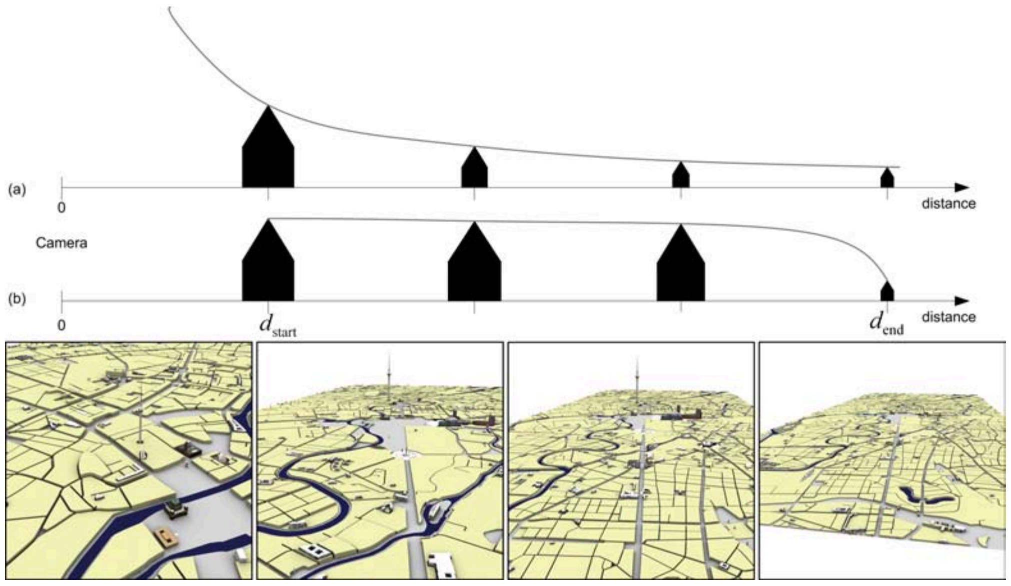

Figure 6: The traditional perspective projection (a) scales objects inversely proportional to the distance. Our scaling (b) keeps the size of landmark object within a distance interval. Note how the dynamic landmarks, e.g., the TV tower, keep their projected size while departing it, until the end distance is reached (screenshots).

While the starting distance $d_{\mathrm{start}} = 2500m$ is kept constant for all landmarks, d_{end} describes, until which distance the landmark should be exaggerated. Therefore it should depend on the size of the reference region of the landmark. Winter et al. use the Voronoi cells as the dual of the landmarks' Delaunay triangulation as reference regions, arguing that reference regions must not overlap. For our requirement of using landmarks as guiding markers, we need them to be visible also outside of the Voronoi cells. Having the landmark hierarchy calculated, the highest layer in which the landmark occurs can be used. In the triangulation, we pick the maximum distance to the vertex' neighboring vertices as d_{end}:

$$d^i_{\mathrm{end}} = \max\{\mathrm{distance}(v_i, n)\ : n \in \mathrm{neighbors}(v_i)\} \tag{2}$$

Though we think this parameterization yields convincing results, it is subjective and not supported by theoretical or empirical research.

5.2 Displacement Handling

Enlarging some elements of the city model consumes space used by other elements. Therefore, the surrounding objects have to be manipulated to avoid intersecting geometry. In [15], we displaced the surrounding non-landmark buildings to avoid these intersections. Here, since the block cells only abstractly represent buildings and are relatively unimportant compared to the dynamic landmarks, we simply clip them against the radial distortion zone of the landmarks. Hence, the space surrounding the enlarged landmark is cleared.

In the case of other dynamic landmarks, clipping of building parts is not possible. Instead, collision between them has to be avoided by displacement. As we have restricted the number of dynamic landmark objects to ten, we can apply a naive brute force displacement: Having calculated the scaling for the dynamic landmarks, we approximate each dynamic landmark by a circle. Then, they are tested pair wise against each other for collision, and, in case of overlap, moved in inverse direction to resolve that single collision. For landmarks having multiple collisions, there will probably still be collisions left. Therefore the collision tests are iterated, until no more overlapping occurs or a maximum number of iterations is reached.

6 Results

6.1 Performance

For our tests, we used a system with an Intel Core 2 processor 6600 (2.4 GHz) with 2 GB RAM and an Nvidia Geforce 8800 graphics card. Our algorithm did not utilize the second core of the CPU. The dataset for Berlin we used consists of app. 60.000 LOD2.5 buildings, 50 CAD models (LOD3-4). Table 6.1 shows the time spent in every stage of the algorithm for processing of generalization levels LOA1-4.

	Time (s)			
	LOA 1	LOA 2	LOA 3	LOA 4
Arrangement	7.44	1.31	0.70	0.55
Point-in-Polygon Tests	5.30	0.38	0.20	0.16
Buffering	19.81	6.83	3.94	2.62
Boolean Set Operations	130.32	50.70	23.80	17.39
Overall (no shadow calculation)	404.80	67.91	32.92	22.91

Table 1: The table shows the measured time to create level of abstraction representations of the input city model consisting of 61.792 polygon features (buildings) and 12.854 line features (infrastructure). The missing time to overall is spent with I/O operations and the statistical calculations.

Concerning the capability to achieve an interactive visualization, we measured satisfying frame rates. Especially, the rendering speed is not bound to the scaling operations and the displacement handling, but rather to the triangle count of the CAD models.

6.2 Problems

While using the infrastructure network as the base for generalization yields an intuitively comprehensible visualization in general, there are detail situations where it fails. For instance, in case of a sparsely built cell with buildings just along one edge, still a block covering the whole cell is created. Here, a solution is needed that further decomposes the cell, until a certain degree of building coverage is reached. Another problem of the current approach is, that the application of Boolean operations sometimes introduces very long and thin elements, which get even worse when they are extruded.

For the highlighting, the displacement handling is sufficient only, if not too many landmarks are colliding. Otherwise, the displacement performs nonintuitive and, for instance, does not preserve the relative angles between the landmarks. Also, as the displacement restarts for every frame, the inter-frame coherence is not enforced, leading to small jumps if the collision situation changes quickly. So far the displacement does not only handle landmarks, while also other constraints such as rivers and roads could be enforced. Technically, another approach is needed if more features have to be dealt with.

7 Conclusion

We present a technique to automatically generalize a virtual 3D city model and create an interactive visualization dynamically highlighting landmark buildings. The technique combines two previously separate techniques and uses a landmark hierarchy to select landmark buildings in smaller scales. The hierarchy proves to ensure a good spatial distribution within the city model, moreover, it yields a decimation of app. $1/3$ between subsequent levels. The distribution also effectively reduces the occurrence of collisions between enlarged global landmarks.

Making global landmarks visible beyond their actual visibility seems to be a good way to provide the user with contextual information. Also, the smooth return to the original size while approaching the landmarks is appealing, as it is predictable. However, it remains as an open question, if the users follow our assumption and actually make use of these reference points. In addition, we have to communicate the fact, that the landmarks are exaggerated, to prevent users from making wrong distance estimations.

In our future, we would like to research ways to blend or morph between several levels of abstraction, addressing the continuous scale of 3D scenes. Therefore we have to deal with transitions between different geometry and handle states between discrete representations. Also we need to adapt the cell based generalization to further refine block cells beyond the granularity of street cells.

TODO.

Acknowledgments

This work has been funded by the German Federal Ministry of Education and Research (BMBF) as part of the InnoProfile research group “3D Geoinformation” (www.3dgi.de).

References

[1] M. Allouche and B. Moulin. Amalgamation in cartographic generalization using Kohonen's feature nets. *International Journal of Geographical Information Science*, 19(8):899–914, 2005.

[2] K.-H. Anders. Level of detail generation of 3d Building groups by aggregation and typification. *Proceedings of 22nd International Cartographic Conference, La Coruña, Spain*, 2005.

[3] M. Barrault, N. Regnauld, C. Duchêne, K. Haire, C. Baeijs, Y. Demazeau, P. Hardy, W. Mackaness, A. Ruas, and R. Weibel. Integrating multi-agent, object-oriented and algorithmic techniques for improved automated map generalization. *Proc. of the 20th International Cartographic Conference*, 3:2110–2116, 2001.

[4] J. Bobrich. *Ein neuer Ansatz zur kartographischen Verdrängung auf der Grundlage eines mechanischen Federmodells*. PhD thesis, Technische Hochschule Darmstadt, 1996.

[5] D. Burghardt and S. Meier. Cartographic Displacement using the Snakes Concept. *Semantic Modeling for the Acquisition of Topographic Information from Images and Maps: SMATI 97*, 1997.

[6] R. Chang, T. Butkiewicz, C. Ziemkiewicz, Z. Wartell, N. Pollard, and W. Ribarsky. Legible simplification of textured urban models. Technical report, University of North Carolina, Charlotte, 2008.

[7] R. Chang, G. Wessel, R. Kosara, E. Sauda, and W. Ribarsky. Legible Cities: Focus-Dependent Multi-Resolution Visualization of Urban Relationships. *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*, 13:1169–1175, 2007.

[8] J. Cohen and D. Manocha. Model Simplification. *Visualization Handbook, Elsevier*, pages 393–411, 2005.

[9] O. G. Consortium. *Candidate OpenGIS CityGML Implementation Specification*, 2006.

[10] V. Coors. Feature-preserving Simplification in Web-based 3D-GIS. *Int. Symp. on Smart Graphics. Proceedings*, 2001.

[11] A. Forberg. Generalization of 3D Building Data based on a Scale Space Approach. *International Archives of Photogrammetry*, 35, 2004.

[12] M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[13] T. Glander and J. Döllner. Cell-based generalization of 3D building groups with outlier management. *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*, 2007.

[14] T. Glander and J. Döllner. Techniques for generalizing building geometry of complex virtual 3d city models. In *2nd International Workshop on 3D Geo-Information: Requirements, Acquisition, Modelling, Analysis, Visualisation*, Delft, Netherlands, December 2007.

[15] T. Glander, M. Trapp, and J. Döllner. A concept of effective landmark depiction in geovirtual 3d environments by view-dependent deformation. In *4th International Symposium on LBS and Telecartography*. Springer, October 2007.

[16] G. Hake, D. Grünreich, and L. Meng. *Kartographie*. Walter de Gruyter, Berlin, New York, 8 edition, 2002.

[17] L. Harrie. The Constraint Method for Solving Spatial Conflicts in Cartographic Generalization. *Cartography and Geographic Information Science*, 26(1), 1999.

[18] P. Heckbert and M. Garland. Survey of polygonal surface simplification algorithms. *SIGGRAPH 97 Course Notes*, 25, 1997.

[19] M. Kada. 3D Building Generalisation. *Proceedings of 22nd International Cartographic Conference, La Coruña, Spain*, 2005.

[20] S. Lamy, A. Ruas, Y. Demazeau, M. Jackson, W. Mackaness, and R. Weibel. The Application of Agents in Automated Map Generalisation. *19th ICA/ACI Conference, Ottawa*, pages 160–169, 1999.

[21] R. McMaster and K. Shea. Generalization in Digital Cartography. *Association of American Geographers*, 1710:20009–3198, 1992.

[22] S. Pion and M. Yvinec. 2d triangulation data structure. In C. E. Board, editor, *CGAL-3.2 User and Reference Manual*. 2006.

[23] J.-Y. Rau, L.-C. Chen, F. Tsai, K.-H. Hsiao, and W.-C. Hsu. Lod generation for 3d polyhedral building model. In *Advances in Image and Video Technology*, pages 44–53, Berlin Heidelberg New York, 2006. Springer-Verlag.

[24] M. Sester. Generalization Based on Least Squares Adjustment. *International Archives of Photogrammetry and Remote Sensing*, 33:931–938, 2000.

[25] F. Thiemann and M. Sester. Segmentation of Buildings for 3D-Generalisation. *Proceedings of the ICA Workshop on Generalisation and Multiple Representation, Leicester, UK*, 2004.

[26] N. Vinson. Design guidelines for landmarks to support navigation in virtual environments. volume 99, pages 278–285, 1999.

[27] J. Ware and C. Jones. Conflict Reduction in Map Generalization Using Iterative Improvement. *GeoInformatica*, 2(4):383–407, 1998.

[28] R. Wein, E. Fogel, B. Zukerman, and D. Halperin. 2d arrangements. In C. E. Board, editor, *CGAL-3.2 User and Reference Manual*. 2006.

[29] S. Winter, M. Tomko, B. Elias, and M. Sester. Landmark hierarchies in context. *Environment and Planning B: Planning and Design advance*, 2008.